NYU Abu Dhabi **Division of Engineering**

ENGR-UH 3511

Computer Organization and Architecture

Name: Nishant Aswani

Net ID: nsa325

Assignment Title: Homework 1

ENGR-UH 3511 Homework 1

Nishant Aswani (nsa325@nyu.edu)

August 31, 2019

Question 1.2

Design for Moore's Law: The idea is similar to **option g** as it suggests accounting for new and upcoming technology in system design.

Use Abstraction to Simplify Design: **option h**, partially relying on existing sensors means that certain technology does not necessarily have to be redesigned.

Making the Common Case Fast: **option d**, express elevators travel to most popular floors, allowing people to skip unnecessary wait time.

Performance via Parallelism: **option b**, because often several suspension bridge cables share the load. It can also refer to dependability via redundancy.

Performance via Pipelining: **option a**, as it suggests specialization and assignment of tasks across a larger task.

Performance via Prediction: **option c**, because these systems carry out predictions based on not necessarily certain weather/wind values.

Hierarchy of Memories: **option e**, because library reserve desks hold recently requested books. Hence, procuring the books when picking up is made much faster, as it is "cached" and easy to access.

Dependability via Redundancy: **option f**, as speed is increased indirectly by increasing gate area, making the transistor more dependable.

Question 1.12.1

Following the idea that the largest clock rate means the best performance, implies that P1 has the better performance.

Using the classic CPU performance equation:

$$\text{P1 CPU Time} = \frac{(5.0E9 \times 0.9)}{4E9} = 1.125s$$
$$\text{P2 CPU Time} = \frac{(1.0E9 \times 0.75)}{3E9} = 0.25s$$

Hence, P2 has the better performance, however, this evaluation relies on all three factors.

Question 1.12.2

$$\text{P1 CPU Time} = \frac{(1.0E9 \times 0.9)}{4E9} = 0.225s$$

We see that P1 requires 0.225s seconds per program. We can now calculate the number of instructions P2 can carry out in this amount of time.

$$\text{P2 Instructions} = \frac{(3E9 \times 0.225)}{0.75} = 0.9E9 \text{ instructions}$$

Hence, P2 can carry out fewer instructions P1 in the same amount of time.

Question 1.12.3

MIPS is an instruction execution rate, and thus it performs inversely to execution time. Thus, a higher MIPS value means faster computation.

$$\text{P1 MIPS} = \frac{5.0E9}{1.125 \times 1E6} = 4.4E3 \text{ MIPS}$$

$$\text{P2 MIPS} = \frac{1.0E9}{0.25 \times 1E6} = 4.0E3 \text{ MIPS}$$

According to the MIPS metric, P1 seems to have a better performance. However, this is counter to the answer found in 1.12.1, where P2 had a better CPU time.

MIPS fails to be a great performance measure as it does not account for instruction count. Hence, variance between programs on the same machine can lead to a wide range of MIPS values.

Question 1.12.4

To calculate MFLOPS for both machines, we assume that 40% of the instructions are floating point operations. We must also recalculate execution time for the FP operations.

$$\text{P1 MFLOPS} = \frac{5.0E9 \times 0.4}{\frac{(5.0E9 \times 0.4 \times 0.9)}{4E9} \times 1E6} = 4.4E3 \text{ MFLOPS}$$

$$\text{P2 MFLOPS} = \frac{1.0E9 \times 0.4}{\frac{(1.0E9 \times 0.4 \times 0.75)}{3E9} \times 1E6} = 4.0E3 \text{ MFLOPS}$$

Question 1.13.1

Reducing the time for FP operations by 20% would result in:

$$0.8 * 70s = 56s \text{ for FP operations, meaning a 14s reduction}$$

a 14s reduction to 236s for the entire program.

Question 1.13.3

Branch instructions are 40s in the 250s program.

$$\frac{40}{250} = 0.16$$

This means that the time for branch instructions makes up 16% of the program time. It is not possible to reduce program time by 20% by simply reducing the time for branch instructions.